|  |  |
| --- | --- |
|  | **Unidev** |
|  | **Unified Development, Inc.**  16690 Swingley Ridge Road  Suite 260  Chesterfield, MO 63017  (636) 532-4424 |

|  |
| --- |
|  |
|  |

**Duncan Solutions, Inc**.

Conditional Fields

April 2014

Revision 1.00

**Prepared by:**

Caleb Miller

Unified Development

**Table of Contents**

[1. Overview 4](#_Toc382218577)

[1.1 Goal 4](#_Toc382218578)

[2. Target Audience 5](#_Toc382218579)

[3. Field Maintenance Services 6](#_Toc382218580)

[3.1 External Services 6](#_Toc382218581)

[3.2 Internal Services 6](#_Toc382218582)

[3.2.1 Deployment 7](#_Toc382218583)

[4. Testing 10](#_Toc382218584)

**Revision** History

|  |  |  |  |
| --- | --- | --- | --- |
| Revision | Author | Description | Revision Date |
| 1.00 | Caleb Miller | Initial version | 04/28/2014 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# Overview

## Goal

We are adding the ability for Duncan admins to specify on a customer by customer basis if certain fields should be hidden in the system. This also applies to the values within filter dropdowns in index pages (like demand area) and if certain filters are displayed at all. NOTE: We are NOT REMOVING any fields from any page, just hiding it from the user via css. The controls and fields still need to be rendered to the browser for the pages to work correctly.

The goal of this document is define how these settings are setup and used from a technical perspective. At the end of the document, the developer should know how to add a new conditional field to the system and be able to update the application to respect that new field.

At the time of this writing, the following fields and filter values will be conditional:

1. Demand Area – Also known as Demand Zone
2. Zone
3. Custom Group 1 – Currently Suburb
4. Custom Group 2 – Currently Not Used
5. Custom Group 3 – Currently Not Used

The following items will be covered:

1. Specifying a conditional field
   1. Define the field
   2. Update customer administration
2. Updating the application to respect the new conditional field
   1. Index page filters
   2. Detail pages
   3. Izenda grid columns
   4. Izenda custom filters

Couple of notes:

1. We are NOT removing the data from the view, just hiding it from the user. The views, stored procs, controller action methods, etc. still all expect this data to be present, even if it is empty. So, we are NOT conditionally adding and removing the fields, filters, etc, but conditionally setting the VISIBILITY of these items
2. For edit or creation pages, sometimes the customer may want to hide a value that is required by the database (Zone for example). They will still be required to create a Default Zone and when adding or editing items that require this field.
   1. If the field is not required, then the application will just hide the column on the edit / create pages.
3. Conditional grid columns that are NOT Izenda related are covered in the Grids Management Document.

# Target Audience

The target audience of this document is person or persons who have:

1. Experience in the following technologies:
   1. C# /.Net /
   2. MVC
   3. T-SQL, SSMS, SqlServer, Stored Procedures and Views
2. The user of this document has a full understanding of the Duncan PEMS project. This includes database and system architecture knowledge, desired business rules of the application, etc.
3. Microsoft SQL Server administration and understand rights, database creation and administration, and are able to use either SSMS or SQL command line interface.
4. Visual Studio 2012

# Specify the Contidional Value

## Definition of the conditional value

The first thing that must occur is to define what field is going to be conditional and the associated rules for that field. There are a few different iterations of rules that will need to be applied to each field, and they are custom per field. Here is an example of differing requirements for two different fields:

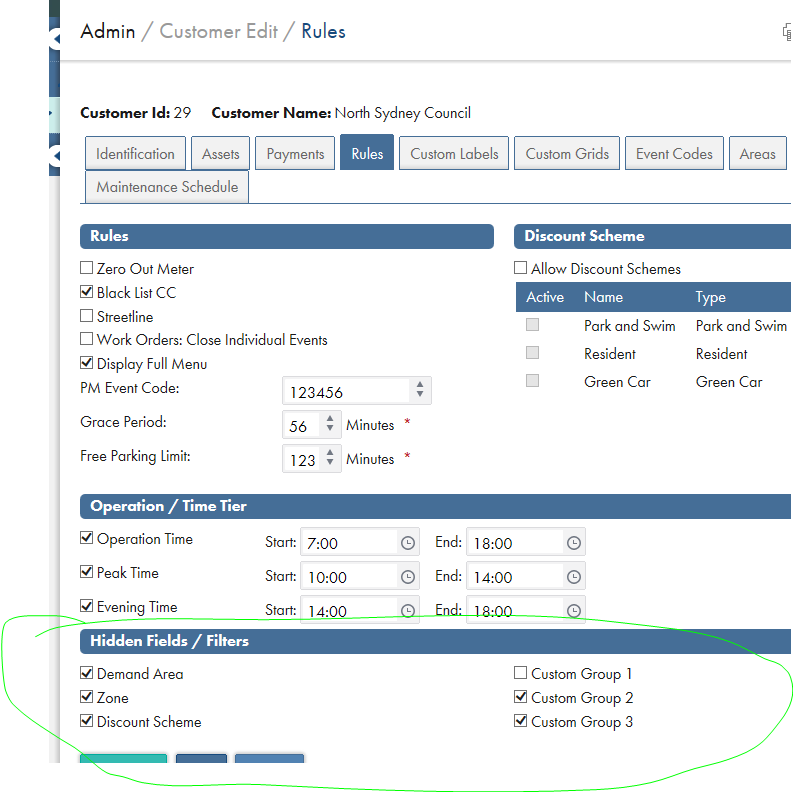
1. Zone – We need to hide zone from the following places
   1. All detail pages in the system
   2. All Grid columns in the system – (Izenda)
   3. Any Index page filter dropdown value in the “Location Types” option
   4. All Izenda Custom Filters
   5. All Izenda Grid Columns
2. Suburb
   1. All Detail pages in the system
   2. All Grid columns in the system – (Izenda)
   3. Any Index page dropdown for Suburbs – the whole thing, not just a value within the drop down.
   4. All Izenda Custom Filters
   5. All Izenda Grid Columns

That being said, these are the steps needed in order to fully define the conditional value:

1. Determine the field that needs to be conditional (Zone)
2. Define the rules for that field (see above)
3. Identify the locations that the field is used within the system (all detail pages, all index filters, grid columns, etc)

## Update Customer Rules

The customer administration “Rules” tab needs to be updated so administrators of the system can define which conditional field that customer needs to display on the site.



Follow the existing examples on how to add these values to the system. This does NOT update the DB structure, as these values are stored with in the customer settings table.

These fields will work just like the “Display full Menu” option, as a customer setting.

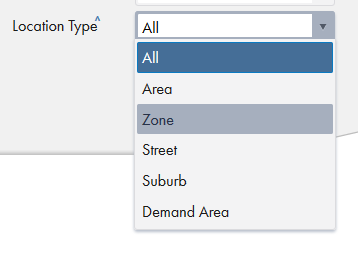
1. The current list of fields to set are:
   1. Demand Zone (also referred to as demand area in some places)
   2. Zone
   3. Customer Group 1 (Currently used for suburb)
   4. Customer Group 2 (currently not used, but adding for future reference)
   5. Customer Group 3 (currently not used, but adding for future reference)
   6. Discount Scheme (some customers do not have discount schemes) and we are NOT using the “Allow discount schemes” flag for this.

# Updating the application

## Index page filters

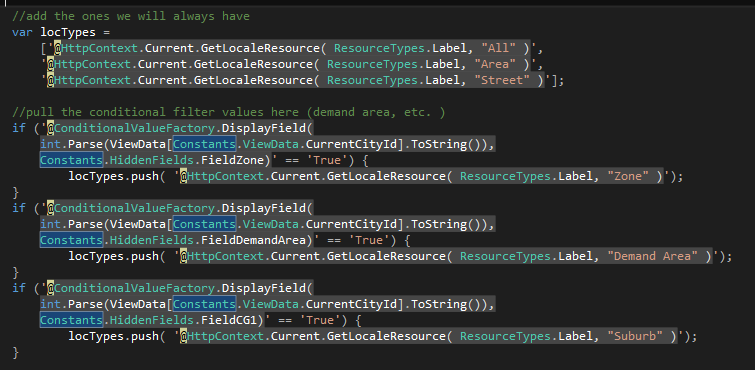
We need to update all of the filters in the index pages to respect our conditional value. There are two basic types of this:

### Filters that reside as options within a drop down list



For these items, we will be conditionally displaying the list items within the drop down list. Each item may have its own rules (don’t show demand area if zones are hidden, etc.), but below is the current code that updates the Asset Index page “Location Types” drop down list for a customer. Everyone will have the “All”, “Area”, and “Street” options; the application just has to respect the conditional values.

We do this with the following example. This code is occurring when we are building the options for our Kendo Observable (MVVM) object on the client. A working example is in the Asset Index page.



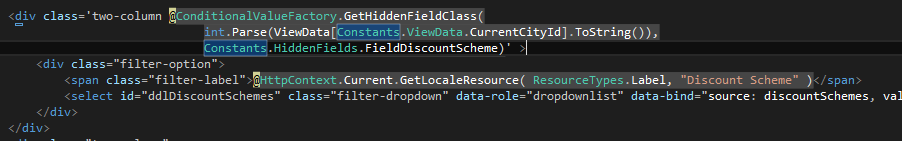
As you can see, we add the items every customer will have, and then conditionally add the other values based on the client selection.

### The entire filter



Some conditional fields have their own dropdown list in the index page filters section, so we will need to hide these as well. Again, we are NOT removing it from the page, as the backed data access requires values from the filters defined. We are just visually hiding this filter to the user.

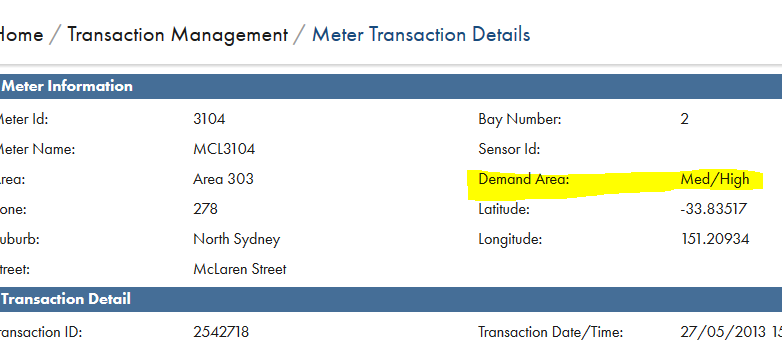
We add a class of “hiddenFilter” to any filter that needs to be hidden. This sets the CSS visibility setting to be “hidden” and sets the display attribute to “none”.



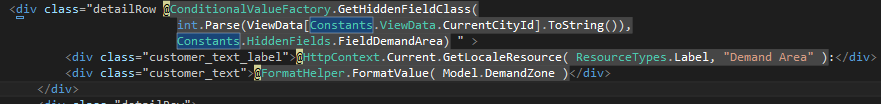
A working example of this is in the Discount Scheme Index page. This is done by calling the GetHiddenFieldClass method of the conditional value factory.

### Detail pages

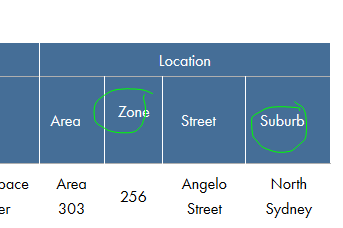
Each reference on the details page for these items needs to do a check against the conditional field factory to denote if the field should be displayed or not. This might cause some display issues of other fields wrapping in to the correct columns, so some re-ordering of the data on the detail pages might be required.



In order to accomplish this, we check the fields required visibility and determine if we should be displaying the field. This is done by calling the GetHiddenFieldClass method in the conditional value factory.



### Izenda grid columns



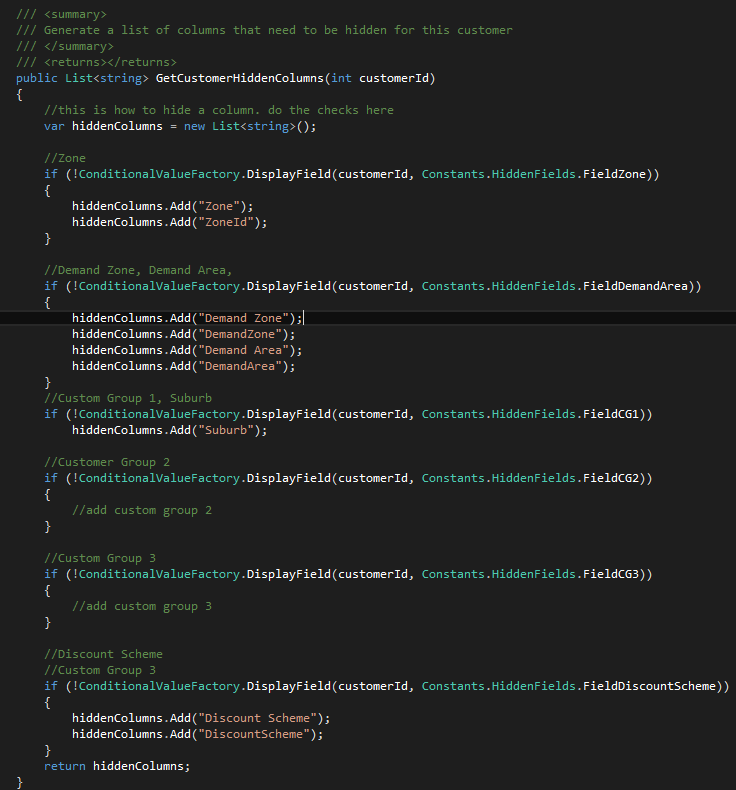
For the Izenda reports, in order to hide a column in the grid we will update the AdHocSettings.HiddenColumns. There is a method in the Izenda Reporting Controller called GetCustomerHiddenColumns that will build a list of all of the columns to hide for a customer.

Each conditional field might have multiple names for its respective column, so we must include all options. Custom Group 1 might be Suburb, etc.

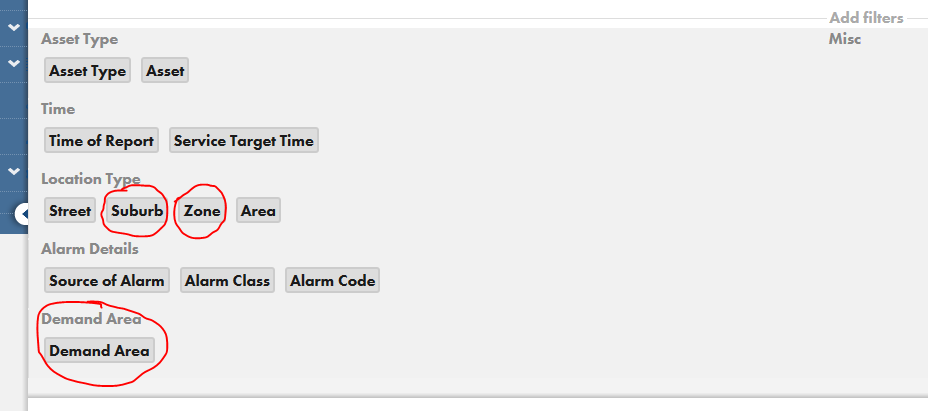
The system is already setup to respect these settings, so adding them in this check is sufficient.



Each time a conditional field is added, this method must be updated to respect that field:



### Izenda custom filters



In order to hide the custom filters in the Izenda reports, we hide them on the client side after they have loaded. What Izenda is doing behind the scenes is not fully known, so instead of not rendering the filter, the application just updates the filters and hides them after they are loaded. The logic that does this respects the Method updated above: GetCustomerHiddenColumns. So if this method is updated correctly, then the filter will be correctly hidden on the client side as well.

The logic that performs the hiding resides on the ReportViewer.cshtml file in /city/Views/Reporting folder. When the document is fully loaded, it rolls through the hidden filters and hides the associated buttons that allows the user to filter by those values.

